Using Machine Learning Models to Predict Option Value

May 5th, 2022

Yuqi Zhang

EXECUTIVE SUMMARY

Purpose of the Project

A European call option gives the owner the right to acquire the underlying security at expiry. For an investor to profit from a call option, the stock's price, at the maturity date, has to be trading high enough above the strike price to cover the cost of the option value.

The purpose of this project is to find the best models to predict the current option values using machine learning algorithms, ultimately helping investors achieve financial safety and success.

Methods to Conduct the Project

The study included a closed examination of four basic factors that affect current option value for the European Call Option, including current asset value (S), strike price of option (K), annual interest rate (r), time to maturity in years (tau).

We also introduced the predicted option value derived from Black-Scholes formula and compared it with our predicted option value using machine learning methods.

Training data was obtained from European call option pricing data on one of the S&P 500 companies, with 1,675 rows and 6 columns (Value, S, K, tau, r, BS). After feature engineering process, we got a dataset with 21 features in total. Then we applied the forward stepwise selection method to rank variables by their multivariate importance. And we tried different models with various combinations of tuning parameters and compared the models using cross-validation.

Results

After trying eight models for regression problem and eight models for classification problem, we chose our best model for each problem by comparing out-of-sample R^2 and accuracy respectively, which is Extreme Gradient Boosting with 99.93% out-of-sample R^2 for regression problem, and Extreme Gradient Boosting with 94.04% accuracy in testing for classification problem.

DATA CLEANING

We performed data cleaning in the following steps:

- There are 5 missing values in 2 rows. Considering we have more than 1,500 records, it is safe to drop both 2 rows
- Then we examined distributions of each feature. There are 2 outliers in tau and 1 outlier in S. Values of outliers are unreasonable, so we decided to drop it
- Besides, there are multiple outliers in r. However, they fall in a reasonable range, and thus we decided to keep them

Therefore, we got a cleaned training dataset with 1,673 records.

FEATURE ENGINEERING

Methodology

While doing the feature engineering, we first merged our training data and hold-out test set together. And then, we generated 15 new variables in the following three ways:

- ***Profitability of the Asset***: we compared current value (S) and strike price of the option (K), and constructed difference indicators to measure our expectation of increase in asset value
- ***Compound Interest Discount***: we utilized the compound interest formula, and calculated risk-free future value of the asset based on current asset value (S) and risk-free interest (r) and expected asset interest rate based on current asset values (S) and strike price of the option (K). Then, we compared those discounted indicators with related practical values to measure the gap between theoretical and practical values
- ***Distribution of Variables***: we examined distributions of original. Specifically, we binned current value (S), strike price of the option (K), and annual interest rate (r) into 5 bins, and generated features to capture extreme values in K and r

Feature Description

Table 1. Variable List

Method	Name	Formula	Comments
Profitability	K-S Ratio	$\frac{K}{S}$	The expected increase of the asset value
	K-S Difference	$K - S$	
	K-S Ratio per Day	$\frac{K}{S \times \tau}$	The expected increase of the asset value per day
Discounted Values	Risk-free FV	$S(1 + r)$	The risk-free future value of the asset
	Risk-free FV Difference	$K - S(1$	The difference between Risk-free FV and K

	Risk-free FV Proportion	$\frac{K-S(1+r)^{\tau}}{K}$	The proportion of difference between risk-free FV and K to K
	Asset Interest Rate	$\frac{K}{S}^{\frac{1}{\tau}-1} =$	The expected interest rate of the asset
	Asset Interest Rate Difference	$r_0 - r$	The difference between Asset Interest Rate and r
	Interest Rate Proportion	$\frac{r_0-r}{r_0}$	The proportion of difference between Asset Interest Rate and r
	Risk-free PV	$\frac{K}{(1+r)^{\tau}}$	The risk-free present value of the asset
Distribution	Quintile of S/ K/ r	\	Approximately equally bin data into 5 parts
	Extremely low K	$\mathrm{I}(K <$ 4	54 out of 1675 observations
	Extremely high r	$\mathrm{I}(r >$ 0.	274 out of 1675 observations

FEATURE SELECTION

After creating variables, we went on a feature selection process. To see the order of multivariate importance, we utilized forward stepwise selection which kept the best variable and then continued to add one more variable, building all possible models and keeping the best new variable at each step. Particularly, we used Sequential Feature Selector, where the scoring was R^2 for the regression problem and accuracy for the classification problem. The following table showed the output.

Table 2. Output of Forward Stepwise Selection

	Regression	**Classification**
add variables in this order	**variable name**	**variable name**
1	risk_free_gap_abs	KS_ratio
2	tau	risk_free_FV
3	risk_free_rate_prop	r_bin
4	risk_free_FV	is_low_K
5	K	S_bin
6	KS_tau_ratio	KS_tau_ratio
7	r_asset	S_expected
8	K_bin	r
9	is_high_r	K_bin
10	r_bin	r_asset
11	r	risk_free_gap_abs
12	S	K
13	is_low_K	risk_free_rate_abs

14	KS_diff	is_high_r
15	risk_free_gap_prop	KS_diff
16	KS_ratio	tau
17	S_bin	S
18	risk_free_rate_abs	risk_free_rate_prop
19	S_expected	risk_free_gap_prop

MODEL ALGORITHMS

Next, we tried several linear and nonlinear models: Linear (Logistic) Regression, Random Forest, Elastic Net, Neural Network, Support Vector Machine, K-Nearest Neighbors, Light Gradient Boosting Machine, and Extreme Gradient Boosting.

Preparation

We first split the training dataset, where 70% was for training and 30% was for testing. Then we ran Linear Regression and Logistic Regression on four basic variables (S, K, r, tau). These were set as the baseline model. The results were included in the appendix.

Model Tuning

In the training set, we tried including different numbers of variables: 3, 5, 10, 15, and 19, based on the order from the forward stepwise selection. We used 5-fold grid search cross-validation that did an exhaustive search over different parameter values for each model. We measured the model performance by R^2 for the regression problem and by accuracy for the classification problem. After that, we used each model with the best parameter settings to predict the outcomes in the designed 30% test set and calculate the out-of-sample R^2 and accuracy. The full list of results was included in the appendix.

Model Selection

Since we compared models with different numbers of variables, we used out-of-sample R^2 as the metric for the regression problem.

The Extreme Gradient Boosting model achieved the highest out-of-sample R^2 and highest accuracy. Thus, we chose it as the final model for both prediction problems. The following tables showed the models and their performance.

Table 3. Final Model for Regression Problem

# of Variables	learning_rate	max_depth	min_child_weight	n_estimators	nthread	objective	Train	Test
5	0.1	5	5	500	4	reg:squarederror	99.89%	99.93%

Table 4. Final Model for Classification Problem

# of Variables	learning_rate	max_depth	min_child_weight	n_estimators	nthread	objective	eval_metric	Train	Test
15	0.1	5	5	100	4	binary:logistic	error	93.35%	94.04%

In the end, we retrained the model based on the entire training dataset and used it to predict the outcomes in the hold-out set, since the more the training data used, the better the performance of the model was.

CONCLUSION

We went through several steps to get the best models. Here's a recapture of each step:

- Checked and dropped outliers and missing values

- Created 15 new variables based on the expertise of option value
- Split the given training data at the ratio of 7 to 3 and saved as training and testing data
- Applied the forward stepwise selection method to rank the variables
- Tried eight different algorithms with various combinations of tuning parameters and chose the best model with the highest out-of-sample R^2 for regression and accuracy for classification problems via cross-validation.
- Trained the models on the whole training dataset with the best models

Our final best model for the regression problem is the Extreme Gradient Boosting (Extreme Gradient Boosting) model with the in-sample R^2 of 99.89% and out-of-sample R^2 of 99.93%. And the best model for the categorical problem is the Extreme Gradient Boosting model with the in-sample accuracy of 93.35% and out-of-sample accuracy of 94.04%

We realized that there were some limitations when building the models and considered improving the models by doing the following future steps:

- Gather more outside data on social, economic, and political events
- Seek advice from domain expertise to optimize the feature engineering process
- Try more algorithms and different combinations of tuning parameters

Business Problems

1. Prediction accuracy outweighs interpretation

In both scenarios, prediction accuracy is more important than interpretation since our final metrics are out-of-sample R^2 and classification error. Also, investors care more about if the model can accurately predict option values so that they can have more accurate information in the trade.

2. Machine learning models outweighs Black-Scholes formula

We chose Machine learning models over Black-Scholes formula because they are impactful in highly dynamic fields such as finance. This financial pricing scheme is non-linear and undergoes a significant amount of calculation to reach a conclusion. However, machine learning models can learn and predict non-linear behavior accurately and catch the new trends when they show up.

3. All four predictor variables should be included

S is the value that the investor gets since he currently acquires the right to purchase the stock, and K is the value the investor will pay in the future. If the investor gets more, he should pay more for Value; if the investor will pay more in the future, he should pay less for Value. When calculating the present value of K, r and tau should also be considered. The higher the r and the longer the tau, the less the investor pays, thus should pay more Value. Thus, all four variables are necessary for prediction.

4. The trained model is not best suited to predict option values for Tesla stocks

Due to the constraints of time and resources, our best models are not sufficient to predict option values for Tesla stocks. Because financial markets are volatile and may depend on political, social, and economic events, which we didn't consider when building the models.

APPENDIX

1. Summary of Features

	count	mean	standard deviation	min	median	max
Value	1,675	15.10	14.04	0.13	11.25	60.15
S	2,795	440.98	7.57	425.47	442.63	455.88
K	2,795	439.09	23.58	375.00	440.00	500.00
tau	2,795	0.20	0.10	0.00	0.20	0.39
r	2,795	0.03	0.00	0.03	0.03	0.03
BS_binary	1,675	0.44	0.50	0.00	0.00	1.00
KS_ratio	2,795	1.00	0.05	0.87	0.99	1.13
KS_diff	2,795	-1.90	23.14	-57.87	-2.46	55.64
risk_free_FV	2,795	443.65	7.53	429.13	445.96	456.57
risk_free_gap_abs	2,795	-4.57	22.92	-61.08	-4.91	51.08
risk_free_gap_prop	2,795	-0.01	0.05	-0.16	-0.01	0.11
KS_tau_ratio	2,795	9.07	16.39	2.28	4.88	253.78
r_asset	2,795	-0.02	0.39	-1.00	-0.03	5.91
risk_free_rate_abs	2,795	-0.05	0.39	-1.03	-0.06	5.88
risk_free_rate_prop	2,795	1.45	18.85	-30.52	1.04	985.42
S_expected	2,795	436.44	23.27	371.60	436.13	495.56
is_low_K	2,795	0.03	0.18	0.00	0.00	1.00
is_high_r	2,795	0.10	0.30	0.00	0.00	1.00

	count	unique	top	top level frequency
S_bin	2,795	5	4	565
K_bin	2,795	5	2	585
r_bin	2,795	5	2	659

2. Models and Performance of the Regression Problem

2.1 Baseline Linear Model

	Train	Test
Linear Regression	90.86%	91.25%

2.2 Linear Model

	# of Variables	Train	Test
Linear Regression	3	92.27%	91.22%
	5	92.26%	91.15%
	10	92.24%	91.44%
	15	98.90%	98.51%
	19	99.29%	99.32%

2.3 Nonlinear Model

Model	# of Variables	Parameters							Train	Test
		max_depth	min_samples_leaf		min_samples_split		n_estimators			
Random Forest	3	10	1		2		90		99.84%	99.89%
	5	30	1		2		80		99.84%	99.89%
	10	25	1		2		90		99.84%	99.89%
	15	25	1		2		70		99.83%	99.87%
	19	10	1		2		30		99.82%	99.86%
		alpha			l1_ratio					
Elastic Net	3	0.00095			0.99				90.83%	91.22%
	5	0.00095			0.99				90.81%	91.20%
	10	0.00095			0.99				90.87%	91.43%
	15	0.0001			0.99				98.33%	98.31%
	19	0.0001			0.99				98.95%	99.06%
		hidden_layer_sizes	activation	solver	alpha	learning_rate	learning_rate_init	max_iter		
Neural Network	3	(30, 30)	logistic	adam	0.001	adaptive	0.001	3000	99.87%	99.87%
	5	(30, 30)	logistic	adam	0.001	invscaling	0.001	3000	99.15%	99.49%
	10	(30, 30)	relu	adam	0.001	constant	0.001	3000	99.67%	99.74%
	15	(20, 20)	relu	adam	0.001	adaptive	0.001	3000	99.57%	99.32%
	19	(20, 20)	relu	adam	0.001	constant	0.001	3000	99.42%	99.50%
		kernel	gamma	C	epsilon		shrinking			
Epsilon-Support Vector Regression	3	rbf	auto	10000	0.01		True		99.77%	99.92%
	5	rbf	auto	100	0.01		True		96.62%	97.08%
	10	rbf	auto	100	0.01		True		96.41%	96.92%
	15	rbf	auto	100	0.01		True		96.77%	97.50%
	19	rbf	auto	100	0.01		True		96.67%	97.19%
		kernel	gamma	C	nu		shrinking			
Nu Support Vector Regression	3	rbf	scale	100	0.5		False		99.64%	99.74%
	5	rbf	scale	10	0.5		False		87.83%	88.79%
	10	rbf	scale	100	0.5		False		87.61%	88.83%
	15	rbf	scale	100	0.5		False		87.48%	88.67%
	19	rbf	scale	100	0.5		False		86.98%	88.27%
		leaf_size			n_neighbors		p			
K-Nearest Neighbor	3	1			5		2		99.58%	-203.43%
	5	1			8		1		98.98%	54.48%
	10	1			3		1		96.94%	-67.04%
	15	1			6		1		97.93%	47.04%
	19	1			4		1		98.30%	-65.53%
		learning_rate	max_depth		min_child_samples		n_estimators			
Light Gradient Boosting Machine	3	0.1	5		10		200		99.83%	99.85%
	5	0.2	5		10		200		99.82%	99.87%
	10	0.2	5		10		200		99.88%	99.90%
	15	0.2	5		10		200		99.88%	99.89%
	19	0.1	20		10		200		99.85%	99.91%
		learning_rate	max_depth	min_child_weight	n_estimators	nthread	objective			
Extreme Gradient Boosting	3	0.1	3	5	500	4	reg:squarederror		99.86%	99.89%
	5	0.1	5	5	500	4	reg:squarederror		99.89%	99.93%
	10	0.1	5	5	500	4	reg:squarederror		99.89%	99.93%
	15	0.1	5	5	500	4	reg:squarederror		99.89%	99.92%
	19	0.1	5	5	500	4	reg:squarederror		99.88%	99.92%

3. Models and Performance of the Classification Problem

3.1 Baseline Linear Model

	Train	Test
Logistic Regression	91.55%	89.86%

3.2 Linear Model

Model	# of Variables	Train	Test
Logistic Regression	3	92.77%	89.07%
	5	91.91%	90.85%
	10	93.59%	91.65%
	15	93.19%	92.25%
	19	92.77%	91.25%

3.3 Nonlinear Model

Model	# of Variables	Parameters							Train	Test
Random Forest		max_depth	min_samples_leaf		min_samples_split		n_estimators			
	3	20	1		2		10		92.92%	92.05%
	5	15	1		5		10		92.83%	92.64%
	10	25	4		5		10		92.75%	93.24%
	15	20	2		10		60		92.75%	93.24%
	19	10	1		5		50		92.58%	93.24%
Elastic Net		alpha	l1_ratio		learning_rate		penalty			
	3	0.0002	0.93		optimal		elasticnet		58.62%	43.54%
	5	0.00035	0.26		optimal		elasticnet		57.42%	43.54%
	10	0.0002	0.91		optimal		elasticnet		91.81%	90.46%
	15	0.00065	0.73		optimal		elasticnet		91.98%	90.66%
	19	0.00095	0.71		optimal		elasticnet		91.81%	89.86%
Neural Network		hidden_layer_sizes	activation	solver	alpha	learning_rate	learning_rate_init	max_iter		
	3	(30, 30)	tanh	adam	0.0001	constant	0.001	600	92.51%	91.29%
	5	(30, 30)	tanh	adam	0.0001	constant	0.001	2000	92.38%	91.73%
	10	(10, 10)	relu	adam	0.0001	constant	0.001	2000	92.78%	92.80%
	15	(10, 10)	relu	adam	0.0001	constant	0.001	2000	92.60%	92.88%
	19	(10, 10)	relu	adam	0.0001	constant	0.001	2000	92.20%	92.50%
Support Vector Machine		kernel	gamma	C	shrinking	probability	class_weight			
	3	rbf	scale	1	True	False	None		93.04%	90.85%
	5	rbf	auto	1	True	False	None		93.05%	90.66%
	10	rbf	scale	1	False	False	None		93.22%	92.64%
	15	rbf	scale	1	False	False	None		93.05%	92.84%
	19	rbf	scale	1	False	False	None		92.42%	92.25%
K-Neares t Neighbor		leaf_size			n_neighbors		p			
	3	1			3		2		93.17%	-129.68%
	5	1			24		2		91.98%	-77.11%
	10	1			19		2		92.66%	-129.68%
	15	1			15		1		92.58%	-129.68%
	19	1			14		1		93.18%	57.14%
Light Gradient Boosting Machine		learning_rate	max_depth		min_child_samples		n_estimators			
	3	0.2	5		30		150		92.58%	91.65%
	5	0.3	5		30		50		93.60%	92.64%
	10	0.2	20		20		50		92.92%	93.64%
	15	0.3	15		20		50		94.11%	92.25%
	19	0.1	5		20		150		94.20%	93.64%
Extreme Gradient Boosting		learning_rate	max_depth	min_child_weight	n_estimators	nthread	eval_metric			
	3	0.2	3	7	500	4	error		93.00%	91.85%
	5	0.2	5	5	500	4	error		93.26%	91.85%
	10	0.1	3	5	300	4	error		93.01%	93.84%
	15	0.1	5	5	100	4	error		93.35%	94.04%
	19	0.1	7	5	100	4	error		93.00%	93.24%